UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/692,515	10/24/2003	Ashish Shah	MSFT-2844/306723.01	9293

41505 7590 11/16/2006

WOODCOCK WASHBURN LLP (MICROSOFT CORPORATION)
ONE LIBERTY PLACE - 46TH FLOOR
PHILADELPHIA, PA 19103

EXAMINER	
SAEED, USMAAN	

ART UNIT	PAPER NUMBER
2166	

DATE MAILED: 11/16/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/692,515	SHAH ET AL.	
	Examiner	Art Unit	
	Usmaan Saeed	2166	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *15 August 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *24 October 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *6/22/2006*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Response to Amendment

1. Applicant's request for reconsideration, filed on 8/15/2006 is acknowledged. Claims 1, 3, 4, 8-10, and 12-30 have been amended.

Claim Objections

2. The amendment to claim 17 has been received and is acceptable to overcome the claim objection.

Claim Rejections - 35 USC § 112

3. The amendments to claims 1, 3, 4, and 8-30 have been received and are acceptable to overcome the 112 second paragraph rejections.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 16-30 overcome the 101 rejections since they recite computer readable storage medium.

Claims 1-30 are still rejected because the claims do not recite a practical application by producing a physical transformation or producing a useful, concrete, and tangible results. To perform a physical transformation, the claimed invention must transform an article of physical object into a different state or thing. Transformation of data is not a physical transformation. A useful, concrete, and tangible results must be either specifically recited in the claim or flow inherently therefrom. To be useful the claimed invention must establish a specific, substantial, and credible utility. To be concrete the claimed invention must be able to produce reproducible results. To be tangible the claimed invention must produce must produce a practical application or real world result.

Information Disclosure Statement

5. Applicants Information Disclosure Statement, filed 6/22/2006, has been received, entered into record and considered. See attached form PTO-1449.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Luoscheng Peng** (**Peng** hereinafter) (U.S. Patent No. 6,317,754) in view of **Oliver Ibelshauser** (**Oliver** hereinafter) (NPL "The WinFS file system for Windows Longhorn: Faster and Smarter" June 17 2003, pages 1-7).

With respect to claim 1, **Peng** teaches **a storage platform system for a hardware/software interface system, implemented at least in part by a computing device, said storage system comprising:**

"multiple instances of a storage platform" as CODA system is applying version vectors to both object replicas and the replication unit that contains set of objects (**Peng** Col 1, Lines 45-47). The Examiner interprets the objects in the reference as instances. **"the storage platform divided into change units"** as the subject system the unit of transmitted data may be a differential update, called an atom because of its small size. This is distinguished from the prior art systems, which must transmit the whole object as the unit of transmitted data (**Peng** Col 4, Lines 2-6).

"a synchronization subsystem native to the hardware/software interface system that enable the system to synchronize the multiple instances of said storage platform" as a system is provided for reliable synchronization of versions of an

object stored at different servers which involves the replacement of either the single central server or a peer-to-peer server system with a network of primary servers linked with high performance reliable links which serve to synchronize secondary servers (**Peng** Col 2, Lines 53-58). Examiner interprets the system as WinFS since it is providing synchronization in peer-to-peer mode. **"based on changes that are sequentially enumerated and tracked on a per change unit basis"** as this is accomplished by either defining a version vector to a whole object or defining a version vector to the base of an object and the update stamp for each of its differential updates (**Peng** Col 3, Lines 33-36).

Peng teaches elements of claim 1 as noted above but does not explicitly teach **"Storage platform such as WinFS and the storage platform divided into change units"**

However, **Oliver** discloses **"Storage platform such as WinFS"** as Windows Future Storage file system will take place in Longhorn, the successor of XP (**Oliver** Page 1) and **"the storage platform divided into change units"** as a cluster is the smallest storage unit on a hard drive. But the sectors are what determines how many Bytes of memory space are physically available for the files. Depending on the partition, you will have one or more sectors of 512 Bytes each in one cluster (**Oliver** Page 2).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to determine the cluster size by file system and the

size of the volume. It would also allow to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

With respect to claim 2, **Peng** teaches **"the system of claim 1 wherein the synchronization subsystem synchronizes only a subset of data, from among the entirety of data on said data store, during a synchronization operation"** as the subject system a summarizing version vector is used to minimize the amount of data transmitted in the synchronizing process by avoiding the necessity for exchanging version vectors for individual objects, whether or not there is any difference in the two objects being synchronized (**Peng** Col 3, Lines 9-14). Examiner interprets the minimized data as subset of data.

With respect to claim 3, **Peng** teaches **"the system of claim 1 wherein a first instance of the storage platform is a replica, that is, running on a hardware/software interface system that has the synchronization subsystem"** as a system is provided for reliable synchronization of versions of an object stored at different servers which involves the replacement of either the single central server or a peer-to-peer server system with a network of primary servers linked with high performance reliable links which serve to synchronize secondary servers (**Peng** Col 2, Lines 53-58). Examiner interprets the system as WinFS since it is providing synchronization in peer-to-peer mode. **"and a second instance of the storage**

platform is a data source, that is, running on a hardware/software interface system that does not have the synchronization subsystem" as CODA system it will be appreciated that it is a file replication system, which does not support peer-to-peer synchronization. It is in essence a client/server system, which will not allow two clients to synchronize directly with each other (**Peng** Col, Lines 48-52).

Peng teaches elements of claim 3 as noted above but does not explicitly teach **"Storage platform such as WinFS & Storage platform such as non-WinFS."**

However, **Oliver** discloses **"Storage platform such as WinFS"** as Windows Future Storage file system will take place in Longhorn, the successor of XP (**Oliver** Page 1) and **"Storage platform such as non-WinFS"** as Fat system (**Oliver** page 2) and NTFS (**Oliver** Page 3).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

With respect to claim 4, **Peng** teaches **"the system of claim 3 wherein the synchronization between the replica and the data source is facilitated by a synchronization adapter that virtualizes the data source by interfacing with an application programming interface of the hardware/software interface system of the replica"** as the system automatically switches between whole object

synchronization and differential synchronization. Further, the subject system permits synchronization between different systems because the semantics of the data is segregated from the synchronization due to extracting updates in a standard format and synchronizing based on a standard protocol (**Peng** Abstract). The system includes a number of object containers 22 and an object container manager 24 coupled thereto. The object container manager is coupled to a synchronizer manager 26, which is in turn coupled to object containers 22 and synchronizers 28. A protocol utility 30 is driven by synchronizer manager to select the most reliable connection to the network. In operation, a system utility or application initiates synchronization from either the object container or the synchronizer manager (**Peng** Col 9, Lines 32-41). Examiner interprets the synchronization adapter as the synchronization manger and synchronizers.

With respect to claim 5, **Peng** teaches **"the system of claim 1 wherein a first pair of instances synchronizes changes independently of a second pair of instances, and wherein both the first pair of instances and the second pair of instances are part of a common sync community"** as the object in a container may be any object, for instance a document, a program, or a row of a table in a relational database, making the subject system a universal system. This integrates the synchronization process for various forms of data and is made possible by the separation of the semantics of objects from the synchronization (**Peng** Col 4, Lines 13-19). Therefore the synchronization is independently done between various forms of data and synchronization is always done between different/pairs of instances.

With respect to claim 6, **Peng** teaches **"the system of claim 1 wherein conflicts in synchronization are automatically detected and resolved based on predefined determinable criteria"** as a method for detecting and resolving conflicts is shown in which a server 180 has a corresponding summarizing version vector (**Peng** Col 12, Lines 11-13). This conflict detection is accomplished by comparing the version vectors or update stamps of the whole object (**Peng** Col 12, Lines 24-26). After the objects 186 and 202 have been found to be in conflict, the conflict is resolved or reconciled by calling a predetermined reconcile method and passing the differential updates in conflict to the method as shown at 220 (**Peng** Col 12, Lines 36-40).

With respect to claim 7, **Peng** teaches **"the system of claim 6 wherein certain of said conflicts are resolved by being logged for manual resolution by an end-user"** as a method for detecting and resolving conflicts is shown in which a server 180 has a corresponding summarizing version vector (**Peng** Col 12, Lines 11-13). Further, since some applications do not permit automatic synchronization, user control of synchronization, which prevents unintended synchronization, is critical (**Peng** Col 8, Lines 2-5).

With respect to claim 8, **Peng** teaches **"the system of claim 1 wherein the synchronization subsystem tracks the state of previous synchronizations with a sync partner, and thereby only synchronizes change units with that partner that**

have changed since the last synchronization" as a server is only concerned with data from a selected number of servers, it is unnecessary to synchronize with all of the servers in the system. In a system which has a large number of servers, only some of which have data which one wishes to synchronize, if one were to attempt to keep track of all objects and all updates, memory would be quickly exhausted (**Peng** Col 4, Lines 61-67). In order to solve this problem in one embodiment of the subject invention, a latest common ancestor version vector is utilized to selectively purge updates and version changes at a selected group of servers which are older than or equal to this latest common ancestor version vector by purging off differential updates or deleted objects which have propagated to the group of the servers in question, e.g. the selected servers (**Peng** Col 5, Lines 1-8).

With respect to claim 9, **Peng** teaches **a method implementing at least in part by a computing device for synchronizing multiple instances of a storage platform for a hardware/software interface systems, said method comprising:**

"Dividing said storage platform into basic units of granularity, change units" as the subject system the unit of transmitted data may be a differential update, called an atom because of its small size. This is distinguished from the prior art systems, which must transmit the whole object as the unit of transmitted data (**Peng** Col 4, Lines 2-6).

"Sequentially enumerating changes and tracking said changes on a per change unit basis" as this is accomplished by either defining a version vector to a

whole object or defining a version vector to the base of an object and the update stamp for each of its differential updates (**Peng** Col 3, Lines 33-36).

"For each instance, tracking the state of changes for that instances, as well as the state of changes for a plurality of other known instances in the sync community" as a server is only concerned with data from a selected number of servers, it is unnecessary to synchronize with all of the servers in the system. In a system which has a large number of servers, only some of which have data which one wishes to synchronize, if one were to attempt to keep track of all objects and all updates, memory would be quickly exhausted (**Peng** Col 4, Lines 61-67). In order to solve this problem in one embodiment of the subject invention, a latest common ancestor version vector is utilized to selectively purge updates and version changes at a selected group of servers which are older than or equal to this latest common ancestor version vector by purging off differential updates or deleted objects which have propagated to the group of the servers in question, e.g. the selected servers (**Peng** Col 5, Lines 1-7).

"For synchronization, identifying new changes by comparing the enumerated changes for a particular instance with the state of changes for that instance" as when synchronization fails, the synchronization will be restored without resending the updates which were already received by the second server in the previous synchronization by comparing the first server's summarizing version vector with the second server's updated version vector (**Peng** Col 3, Lines 62-67).

Peng teaches elements of claim 9 as noted above but does not explicitly teach, **"Dividing said storage platform into basic units of granularity & "Storage platform such as Win FS"."**

However, **Oliver** discloses, **"Dividing said storage platform into basic units of granularity"** as a cluster is the smallest storage unit on a hard drive. But the sectors are what determines how many Bytes of memory space are physically available for the files. Depending on the partition, you will have one or more sectors of 512 Bytes each in one cluster (**Oliver** Page 2) and **"Storage platform such as WinFS"** as Windows Future Storage file system will take place in Longhorn, the successor of XP (**Oliver** Page 1).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to determine the cluster size by file system and the size of the volume. It would also allow to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

With respect to claim 10, **Peng** teaches **"the method of claim 9, wherein a first instance, a replica, is instantiated on a hardware/software interface system that directly supports Item-based synchronization"** as a system is provided for reliable synchronization of versions of an object stored at different servers which involves the replacement of either the single central server or a peer-to-peer server system with a

network of primary servers linked with high performance reliable links which serve to synchronize secondary servers (**Peng** Col 2, Lines 53-58). Examiner interprets the system as WinFS since it is providing synchronization in peer-to-peer mode. **"and wherein a second instance, a data source, is instantiated on a hardware/software interface system that does not directly support Item-based synchronization"** as CODA system it will be appreciated that it is a file replication system, which does not support peer-to-peer synchronization. It is in essence a client/server system, which will not allow two clients to synchronize directly with each other (**Peng** Col, Lines 48-52). **"said method further comprising the use of an adapter to virtualize the second instance via a synchronization application programming interface"** as a system utility or application initiates synchronization from either the object container or the synchronizer manager. Synchronizer manager 26 consults with utility 30 to open a reliable connection between two servers to be synchronized. Thereafter, synchronizer manger 26 creates a synchronizer such as synchronizer 28 based on the result from the protocol utility. Then the synchronizers on the two servers will initiate the synchronization process (**Peng** Col 9, Lines 39-46). The system automatically switches between whole object synchronization and differential synchronization. Further, the subject system permits synchronization between different systems because the semantics of the data is segregated from the synchronization due to extracting updates in a standard format and synchronizing based on a standard protocol (**Peng** Abstract).

Peng teaches elements of claim 10 as noted above but does not explicitly teach **"Storage platform such as WinFS & Storage platform such as non-WinFS."**

However, **Oliver** discloses **"Storage platform such as WinFS"** as Windows Future Storage file system will take place in Longhorn, the successor of XP (**Oliver** Page 1) and **"Storage platform such as non-WinFS"** as Fat system (**Oliver** page 2) and NTFS (**Oliver** Page 3).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

With respect to claim 11, **Peng** teaches **"the method of claim 10 further comprising detecting synchronization conflicts at the level of change unit granularity"** as a method for detecting and resolving conflicts is shown in which a server 180 has a corresponding summarizing version vector (**Peng** Col 12, Lines 11-13).

Peng teaches elements of claim 11 as noted above but does not explicitly teach, **"change unit granularity."**

However, **Oliver** discloses, **"change unit granularity"** as a cluster is the smallest storage unit on a hard drive. But the sectors are what determines how many Bytes of memory space are physically available for the files. Depending on the partition, you will have one or more sectors of 512 Bytes each in one cluster (**Oliver** Page 2).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to determine the cluster size by file system and the size of the volume. It would also allow to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

With respect to claims 12, **Peng** teaches **"the method of claim 10, further comprising: instances reporting success, failure, and/or conflicts at individual change unit level on change application, the instance comprising sync data"** as a method for detecting and resolving conflicts is shown in which a server 180 has a corresponding summarizing version vector (**Peng** Col 12, Lines 11-13). The version vector of the corresponding object and the summarizing version vector in the second sever will be updated right after it successfully receives the update. Therefore, when synchronization fails, the synchronization will be restored without resending the updates which were already received by the second server in the previous synchronization by comparing the first server's summarizing version vector with the second server's updated version vector (**Peng** Col 3, Lines 59-67).

"applications using sync data for updating a backend state" as the term summarizing version vector as used herein means a vector having fields, which summarize the state of the object container at a server. Each summarizing version vector is a vector of update stamps. Each update stamp has a field for the associated

object container's identifier and a field for the associated time stamp (**Peng** Col 3, Lines 15-20).

Peng teaches elements of claim 12 as noted above but does not explicitly teach, **"Unit level of change."**

However, **Oliver** discloses, **"Unit level of change"** as a cluster is the smallest storage unit on a hard drive. But the sectors are what determines how many Bytes of memory space are physically available for the files. Depending on the partition, you will have one or more sectors of 512 Bytes each in one cluster (**Oliver** Page 2).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to determine the cluster size by file system and the size of the volume. It would also allow to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

With respect to claim 13, **Peng** teaches **a method implemented at least in part by a computing device for synchronizing a replica with a data source, each being a sync partner, wherein both said replica and said data source have change state information that is maintained by each sync partner, and wherein said data source uses an adapter to interface with a hardware/software interface system of said replica, said method comprising:**

"Said replica sending to said adapter an updated state information for said replica that, based on a last state information for said data source, reflect new changes that have been made since the last synchronization as reflected in said last state information for said data source" as the term summarizing version vector as used herein means a vector having fields, which summarize the state of the object container at a server. Each summarizing version vector is a vector of update stamps. Each update stamp has a field for the associated object container's identifier and a field for the associated time stamp (**Peng** Col 3, Lines 15-20). Object container 120 is changed so that it contains synchronizing information supplied by summarizing version vector 1 so that it in turn updates object container 110 throughout information sent as illustrated by arrow 124 (**Peng** Col, Lines 53-57).

"Said adapter, receiving said updated state information for said replica and said new changes, implementing as many changes to the data source as possible and tracking success or failure for each change on a change unit by change unit basis" as the version vector of the corresponding object and the summarizing version vector in the second sever will be updated right after it successfully receives the update. Therefore, when synchronization fails, the synchronization will be restored without resending the updates which were already received by the second server in the previous synchronization by comparing the first server's summarizing version vector with the second server's updated version vector (**Peng** Col 3, Lines 59-67). **"wherein changes are sequentially enumerated and tracked on a per change unit basis"** as this is accomplished by either defining a version vector to a whole object or defining a

version vector to the base of an object and the update stamp for each of its differential updates (**Peng** Col 3, Lines 33-36).

Peng teaches elements of claim 13 as noted above but does not explicitly teach, **"Change unit basis, Storage platform such as WinFS and Storage platform such as non-WinFS."**

However, **Oliver** discloses, **"Change unit basis"** as a cluster is the smallest storage unit on a hard drive. But the sectors are what determines how many Bytes of memory space are physically available for the files. Depending on the partition, you will have one or more sectors of 512 Bytes each in one cluster (**Oliver** Page 2), **"Storage platform such as WinFS"** as Windows Future Storage file system will take place in Longhorn, the successor of XP (**Oliver** Page 1) and **"Storage platform such as non-WinFS"** as Fat system (**Oliver** page 2) and NTFS (**Oliver** Page 3).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to determine the cluster size by file system and the size of the volume. It would also allow to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

With respect to claim 14, **Peng** teaches **"the method of claim 13, further comprising: said adapter calculating the new state of the data source based on the success or failure for each change on a change unit by change unit basis,**

storing this new state information, and transmitting this new state information to the hardware/software interface system of the replica said hardware/software interface system of the replica storing said new state information for said data source for future use by said replica" as check if any of these objects' base version vector is newer than the common version vector of the two servers, where base version vector of an object refers to the version vector of the object absent any differential updates and where the common version vector refers to a version vector reflecting the state from where the two servers' summarizing version vectors diverged (**Peng** Col 5, Lines 61-67). The second server will store or update the first server's summarizing version vector it has stored previously. It may also recalculate its latest common ancestor version vector if all of the selected server's summarizing version vectors have been stored in the second server and accordingly purges off all the deleted object's information or differential updates whose version vectors or update stamps are older than or equal to the latest common ancestor version vector (**Peng** Col 6, Lines 41-50). FIG. 8 is a block diagram of the system for extracting updates to be transmitted from a first server to a second server utilizing summarizing version vectors and a differential update log for the second server (**Peng** Col 8, Lines 64-67).

Peng teaches elements of claim 14 as noted above but does not explicitly teach **"Storage platform such as WinFS."**

However, **Oliver** discloses **"Storage platform such as WinFS"** as Windows Future Storage file system will take place in Longhorn, the successor of XP (**Oliver** Page 1).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

With respect to claim 15, **Peng** teaches **the method of claim 13 further comprising:**

"said adapter transmitting to the hardware/software interface system of the replica the success or failure for each change on a change unit by change unit basis" as the version vector of the corresponding object and the summarizing version vector in the second sever will be updated right after it successfully receives the update. Therefore, when synchronization fails, the synchronization will be restored without resending the updates which were already received by the second server in the previous synchronization by comparing the first server's summarizing version vector with the second server's updated version vector (**Peng** Col 3, Lines 59-67).

"said hardware/software interface system of the replica calculating a new state information for the data source based on the success or failure for each change to the data source on a change unit by change unit basis" " as check if any of these objects' base version vector is newer than the common version vector of the two servers, where base version vector of an object refers to the version vector of the object absent any differential updates and where the common version vector refers to a

version vector reflecting the state from where the two servers' summarizing version vectors diverged (**Peng** Col 5, Lines 61-67). The second server will store or update the first server's summarizing version vector it has stored previously. It may also recalculate its latest common ancestor version vector if all of the selected server's summarizing version vectors have been stored in the second server and accordingly purges off all the deleted object's information or differential updates whose version vectors or update stamps are older than or equal to the latest common ancestor version vector (**Peng** Col 6, Lines 41-50).

"said hardware/software interface system of the replica transmitting the new state information to the adapter and storing said new state information for future use by said replica; and said adapter receiving and storing said new state information" FIG. 8 is a block diagram of the system for extracting updates to be transmitted from a first server to a second server utilizing summarizing version vectors and a differential update log for the second server (**Peng** Col 8, Lines 64-67). Examiner interprets the synchronizer 28 on the servers as adapter.

Peng teaches elements of claim 15 as noted above but does not explicitly teach, **"Change unit basis, and Storage platform such as WinFS."**

However, **Oliver** discloses, **"Change unit basis"** as a cluster is the smallest storage unit on a hard drive. But the sectors are what determines how many Bytes of memory space are physically available for the files. Depending on the partition, you will have one or more sectors of 512 Bytes each in one cluster (**Oliver** Page 2), **"Storage**

platform such as WinFS" as Windows Future Storage file system will take place in Longhorn, the successor of XP (**Oliver** Page 1).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to determine the cluster size by file system and the size of the volume. It would also allow to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

Claims 16-30 are essentially the same as claims 1-15 except they set forth the claimed invention as a computer-readable medium comprising instructions and are rejected for the same reason as applied hereinabove.

Response to Arguments

7. Applicant's arguments with respect to independent claims 1, 9, 13, 16, 24, and 28, which recite similar features have been fully considered but they are not persuasive.

Applicant argues that **Peng** does not teach, **"the storage platform divided into change units" and "synchronization based on changes that are sequentially enumerated and tracked on a per change unit basis."**

In response to the preceding arguments, Examiner respectfully submits that **Peng** teaches, **"the storage platform divided into change units"** as the subject system the unit of transmitted data may be a differential update, called an atom because of its small size. This is distinguished from the prior art systems, which must transmit the whole object as the unit of transmitted data (**Peng** Col 4, Lines 2-6).

"based on changes that are sequentially enumerated and tracked on a per change unit basis" as this is accomplished by either defining a version vector to a whole object or defining a version vector to the base of an object and the update stamp for each of its differential updates (**Peng** Col 3, Lines 33-36).

Peng teaches elements of claim 1 as noted above but does not explicitly teach **"Storage platform such as WinFS and the storage platform divided into change units"**

However, **Oliver** discloses **"Storage platform such as WinFS"** as Windows Future Storage file system will take place in Longhorn, the successor of XP (**Oliver** Page 1) and **"the storage platform divided into change units"** as a cluster is the smallest storage unit on a hard drive. But the sectors are what determines how many Bytes of memory space are physically available for the files. Depending on the partition, you will have one or more sectors of 512 Bytes each in one cluster (**Oliver** Page 2).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of the cited references because **Oliver's** teachings would have allowed **Peng** to determine the cluster size by file system and the

size of the volume. It would also allow to store files based on various content criteria e.g. author, content, names, sources medium and the most recent user and provide a FS which is entirely based on a relational database.

The claimed change units are not described explicitly enough in the claims to overcome the rejection and could be interpreted in different ways. The cited prior art still reads on the claimed change units. Further the synchronization between individual objects in Peng could also be interpreted as change units.

Peng teaches in file systems, a directory can be considered as a file group and it may further define structures inside to organize data (files). In object-oriented databases, a data group possibly includes all the homogeneous or heterogeneous objects needed to meaningfully support its applications. As the first step, we assume that all the data groups in different devices, which need to be synchronized with each other, eventually contain exactly the same data set (**Peng** Col 13, Lines 33-42).

Conclusion

8. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not

mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Contact Information

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Usmaan Saeed whose telephone number is (571)272-4046. The examiner can normally be reached on M-F 8-5.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Hosain Alam can be reached on (571)272-3978. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Usmaan Saeed

Patent Examiner
Art Unit: 2166

Leslie Wong
Primary Examiner
LW

US
November 06, 2006

HOSAIN ALAM
SUPERVISORY PATENT EXAMINER